UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

Parcial II – Parte Práctica (40%)

Materia: Inteligencia Artificial aplicada a la Ciberseguridad **Profesor**: Dr. Carlos A. Rovetto
Fecha: lunes, 9 de junio de 2025 Grupo: 1S3131

Estudiante: Omar Polanco Cédula: 8-1014-120

Estudiante: Yul Pinto Cédula: 8-885-453

Objetivo
Aplicar los conocimientos recibidos sobre la creación de datasets utilizando Python y el análisis de estos.

Descripción: Crear un dataset desde procesos de su computadora y subirlo a un sitio público como los siguientes (GitHub, Google Drive, Kaggle, AWS S3, Dropbox, etc). Posteriormente, elabore el código Python para acceder a él a través de python y genere la siguiente información para el dataset generado.

- Estadísticas básicas: Media, mediana, moda y desviación estándar.
- Correlación entre variables: Matriz de correlación para ver las relaciones entre variables numéricas.

Ejemplos de tipos de datasets que puede crear

1. Uso de la CPU: Detectar anomalías en el comportamiento de la CPU para identificar posibles ataques de denegación de servicio (DDoS).
2. Uso de la Memoria RAM: Monitorear el uso excesivo de memoria para detectar malware o procesos maliciosos.
3. Uso del Disco Duro: Identificar cambios sospechosos en el uso del disco que podrían indicar filtración de datos o actividad maliciosa.
4. Uso de la Red: Detectar patrones de tráfico anómalos para identificar intrusiones o ataques de red.
5. Lista de Procesos Activos: Clasificar procesos maliciosos y legítimos para prevenir accesos no autorizados o malware.
6. Temperatura de Componentes del Sistema: Detectar sobrecalentamientos anómalos que puedan indicar un ataque físico o manipulación del hardware.
7. Tiempo de Actividad del Sistema: Analizar tiempos de actividad inusuales para detectar sistemas comprometidos o accesos no autorizados.
8. Eventos del Sistema (Logs): Identificar patrones de eventos que puedan señalar intentos de intrusión o fallos de seguridad.
9. Consumo de Energía: Detectar patrones inusuales de consumo energético relacionados con ataques físicos o manipulaciones de hardware.
10. Actividades de Entrada del Usuario (Teclado): Analizar patrones de comportamiento del usuario para detectar accesos no autorizados o actividades maliciosas.

Observaciones: Se le dará un enlace de un formulario para que suba el código Python y el enlace del dataset público. Debe asegurarse que a través de la ejecución del código se puede tener acceso al dataset.

Resultados:
Creacion de Dataset:

```python
import pandas as pd
import numpy as np
from datetime import datetime, timedelta

# Configuración para la generación de datos
np.random.seed(42)
num_registros = 1000
componentes = ['CPU', 'GPU', 'RAM', 'SSD', 'Motherboard']

# Generar timestamps (últimos 7 días, cada ~10 minutos)
timestamps = [datetime.now() - timedelta(days=7) + timedelta(minutes=10*i) for i in
range(num_registros)]

# Generar datos
data = {
    'timestamp': timestamps,
    'component_id': np.random.choice(componentes, num_registros),
    'temperatura': np.concatenate([
        np.random.normal(40, 5, 900),  # Datos normales
        np.random.normal(80, 10, 50),   # Picos de temperatura
        np.random.normal(70, 5, 50)    # Sobrecalentamiento sostenido
    ]),
    'normal_temp_min': np.random.choice([30, 35, 40, 25, 45], num_registros),
    'normal_temp_max': np.random.choice([60, 65, 70, 55, 75], num_registros)
}

# Crear DataFrame
df = pd.DataFrame(data)

# Asegurar que temp_min < temp_max
df['normal_temp_min'] = np.minimum(df['normal_temp_min'], df['normal_temp_max'] - 5)
df['normal_temp_max'] = np.maximum(df['normal_temp_max'], df['normal_temp_min'] + 5)

# Determinar anomalías
df['is_anomalo'] = (df['temperatura'] < df['normal_temp_min']) | (df['temperatura'] >
df['normal_temp_max'])
df['tipo_anomalia'] = '-'
df.loc[(df['is_anomalo']) & (df['temperatura'] > df['normal_temp_max'] + 15), 'tipo_anomalia'] =
'spike'
df.loc[(df['is_anomalo']) & (df['temperatura'] > df['normal_temp_max'] + 5) & (df['temperatura'] <=
df['normal_temp_max'] + 15), 'tipo_anomalia'] = 'sustained_overheat'

# Guardar dataset
df.to_csv('temperatura_componentes.csv', index=False)

print("Dataset generado exitosamente!")
print(f"Total de registros: {len(df)}")
print(f"Anomalías detectadas: {df['is_anomalo'].sum()}")
```

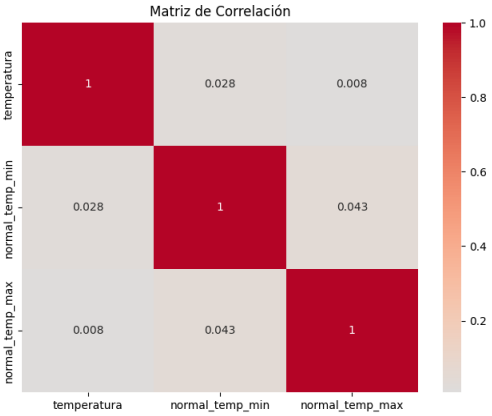
Este dataset simula mediciones de temperatura de componentes clave de una computadora (CPU, GPU, RAM, SSD, etc.), registrando valores térmicos junto con sus rangos normales operativos. Incluye detección automática de anomalías (como picos repentinos o sobrecalentamiento sostenido), lo que permite analizar patrones de comportamiento térmico y posibles fallos. Los datos son ideales para practicar análisis exploratorios, detección de outliers y correlaciones entre variables, con aplicaciones en mantenimiento predictivo y optimización de sistemas.

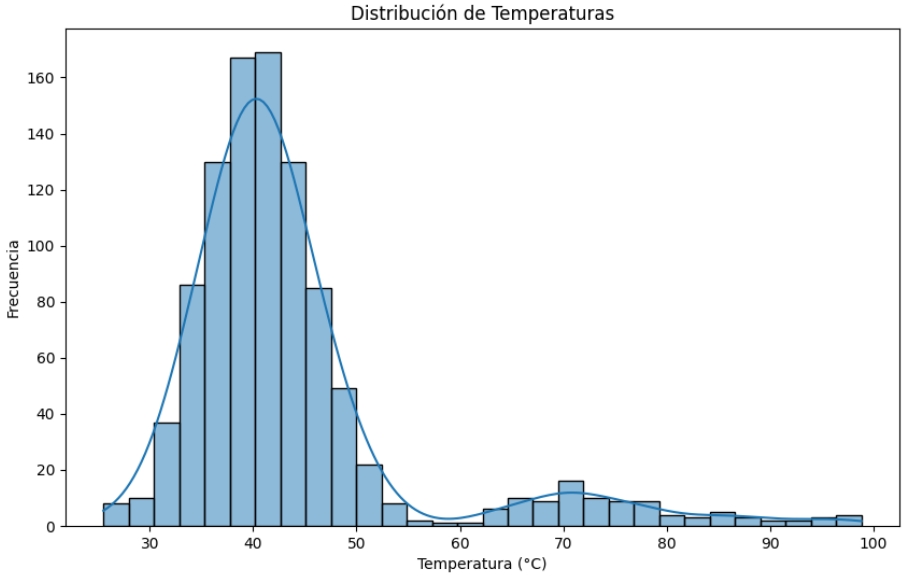
Analisis de Dataset:

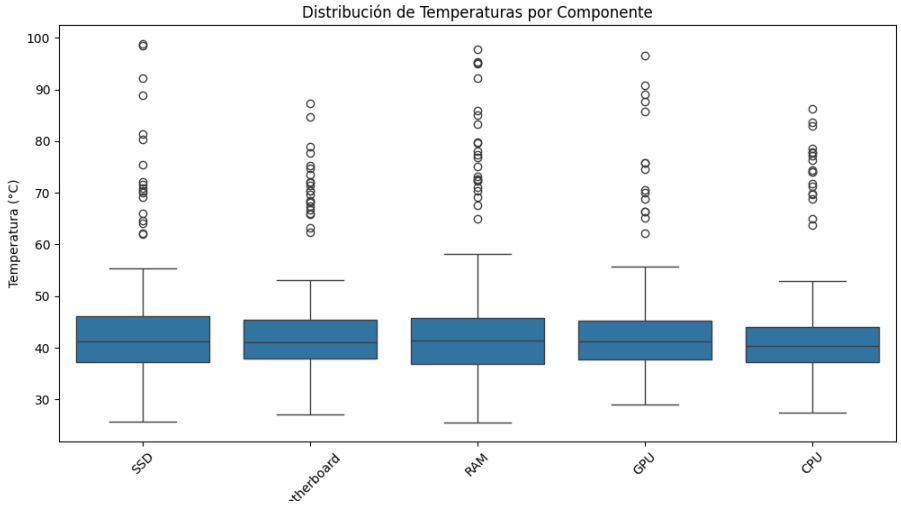
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os

# Configuración para mostrar gráficos en VS Code
plt.switch_backend('TkAgg')  # O usa 'Qt5Agg' si tienes problemas

# 1. Verificar existencia del archivo
archivo_csv = 'temperatura_componentes.csv'
if not os.path.exists(archivo_csv):
    print(f"Error: No se encontró el archivo {archivo_csv}")
    print("Por favor ejecuta primero el script de generación de datos")
    exit()

# 2. Cargar los datos
try:
    df = pd.read_csv(archivo_csv)
    print("Dataset cargado exitosamente!")
    print(f"Total de registros: {len(df)}")
except Exception as e:
    print(f"Error al cargar el dataset: {e}")
    exit()

# 3. Análisis básico
print("\n=== ESTADÍSTICAS BÁSICAS ===")
print(df.describe())

# 4. Estadísticas por componente
print("\n=== ESTADÍSTICAS POR COMPONENTE ===")
stats_componentes = df.groupby('component_id').agg({
    'temperatura': ['mean', 'median', 'std', 'max'],
    'is_anomalo': 'sum'
})
print(stats_componentes)

# 5. Análisis de anomalías
print("\n=== ANÁLISIS DE ANOMALÍAS ===")
anomalias = df[df['is_anomalo']]
print(f"Total anomalías: {len(anomalias)} ({len(anomalias)/len(df)*100:.2f}%)")
print("\nTipos de anomalía:")
print(anomalias['tipo_anomalia'].value_counts())

# 6. Visualizaciones
print("\nGenerando visualizaciones...")

# Gráfico 1: Distribución de temperaturas
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='temperatura', bins=30, kde=True)
plt.title('Distribución de Temperaturas')
plt.xlabel('Temperatura (°C)')
plt.ylabel('Frecuencia')
plt.show()

# Gráfico 2: Temperaturas por componente
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='component_id', y='temperatura')
plt.title('Distribución de Temperaturas por Componente')
plt.xlabel('Componente')
plt.ylabel('Temperatura (°C)')
plt.xticks(rotation=45)
plt.show()

# Gráfico 3: Matriz de correlación
numeric_cols = ['temperatura', 'normal_temp_min', 'normal_temp_max']
plt.figure(figsize=(8, 6))
sns.heatmap(df[numeric_cols].corr(), annot=True, cmap='coolwarm', center=0)
plt.title('Matriz de Correlación')
plt.show()

print("\n¡Análisis completado con éxito!")
```

Link en GitHub:
https://github.com/omarp18/parcial2-dataset#

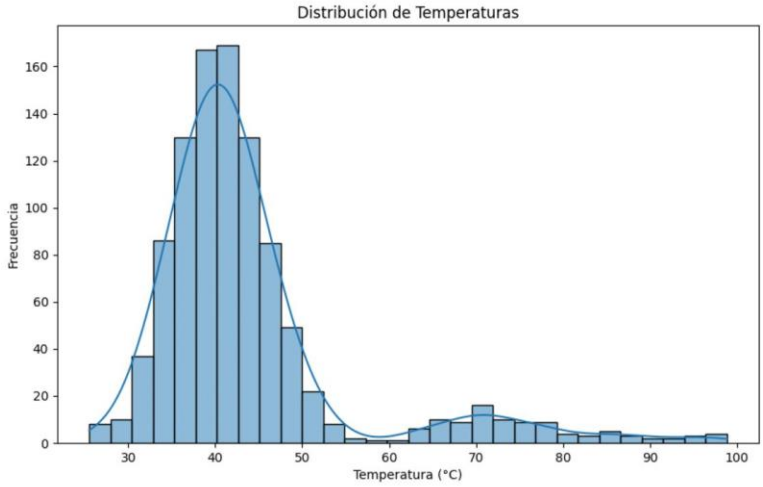
Imagenes de Analisis:
Resultados de Omar:

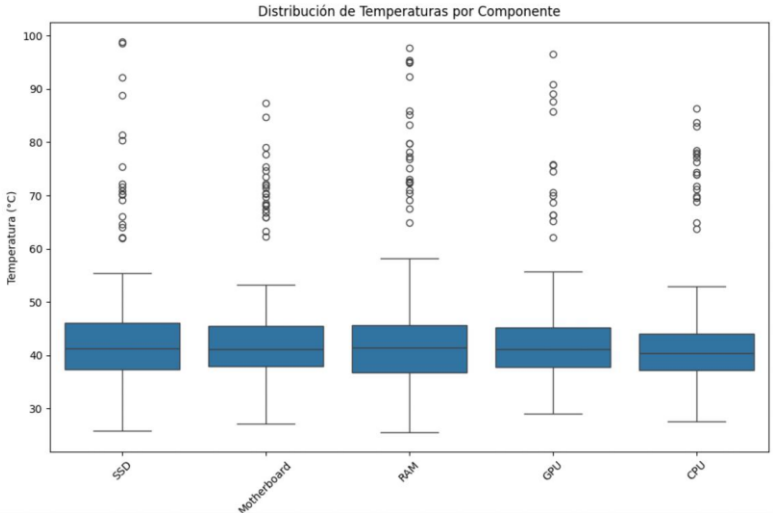


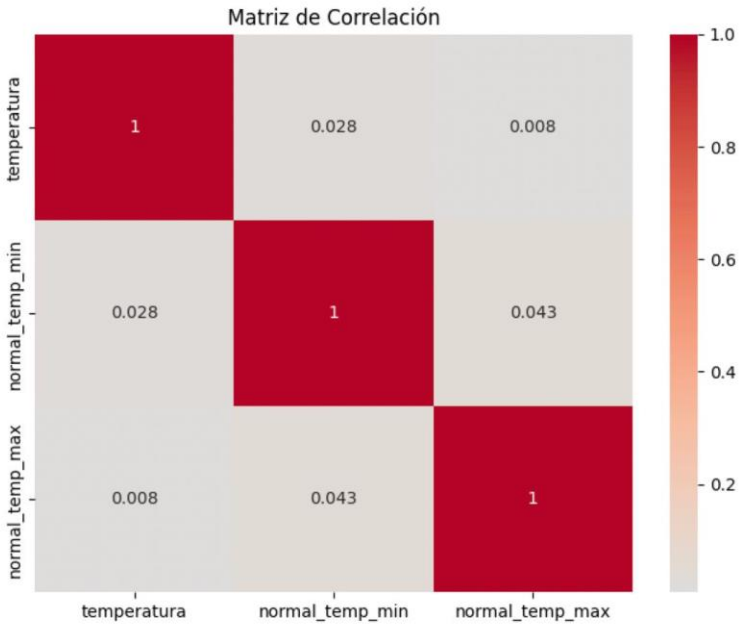




Resultados de Jouls:







Dataset Creado:

1	timestamp	component_id	temperatura	normal_temp_min	normal_temp_max	is_anomalo	tipo_anomalia
2	2025-06-02 07:38:09.054105	SSD	40.546973973024464	25	60	False	-
3	2025-06-02 07:48:09.054114	Motherboard	43.62883311949346	30	65	False	-
4	2025-06-02 07:58:09.054117	RAM	42.40504615868357	40	75	False	-
5	2025-06-02 08:08:09.054119	Motherboard	41.119420121395656	35	60	False	-
6	2025-06-02 08:18:09.054121	Motherboard	36.04762772277344	40	75	True	-
7	2025-06-02 08:28:09.054123	GPU	42.35734178567998	40	55	False	-
8	2025-06-02 08:38:09.054125	RAM	49.41012248237517	45	75	False	-
9	2025-06-02 08:48:09.054127	RAM	46.72710023077489	40	70	False	-
10	2025-06-02 08:58:09.054129	RAM	47.965933133196984	40	70	False	-
11	2025-06-02 09:08:09.054130	Motherboard	37.443921617844076	25	75	False	-
12	2025-06-02 09:18:09.054132	SSD	35.0519758987071	35	70	False	-
13	2025-06-02 09:28:09.054134	RAM	39.37106539950176	25	55	False	-
14	2025-06-02 09:38:09.054135	Motherboard	40.27862456144347	25	65	False	-
15	2025-06-02 09:48:09.054137	GPU	45.47095759235474	30	70	False	-
16	2025-06-02 09:58:09.054139	SSD	31.537676851425903	25	70	False	-
17	2025-06-02 10:08:09.054140	GPU	47.64775159730307	45	75	False	-
18	2025-06-02 10:18:09.054142	SSD	39.20996050710505	45	55	True	-
19	2025-06-02 10:28:09.054144	Motherboard	37.8655946502629	30	65	False	-
20	2025-06-02 10:38:09.054145	CPU	34.93947812369916	35	75	True	-
21	2025-06-02 10:48:09.054147	SSD	31.725716640671152	35	75	True	-
22	2025-06-02 10:58:09.054149	GPU	44.11585291980957	25	55	False	-
23	2025-06-02 11:08:09.054150	Motherboard	40.36658983594202	25	65	False	-
24	2025-06-02 11:18:09.054152	SSD	33.55019550129473	25	55	False	-